JavaScript et HTML

Cours 9

Jean-Jacques Lévy

jean-jacques.levy@inria.fr

http://jeanjacqueslevy.net/lp-js

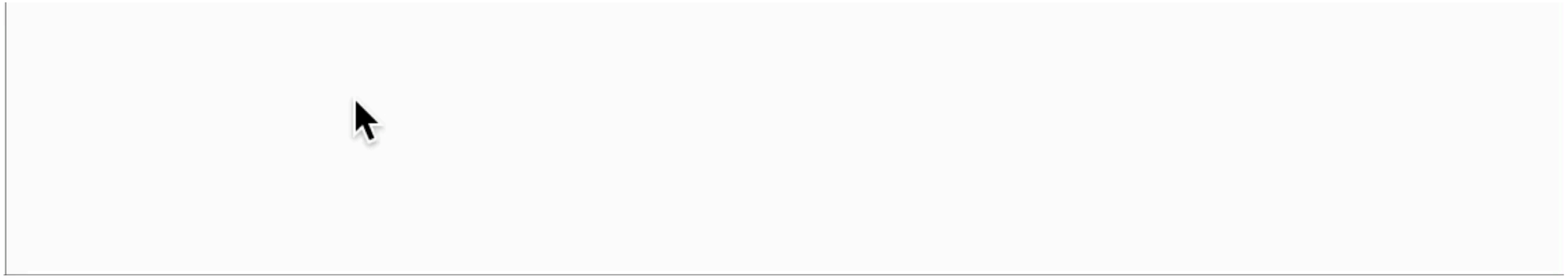
Plan

- solution des exercices
- exemples de scripts CGI
- HTML++
- w3-css

Solutions des exercices

Exercice 801 Faire un texte d'alerte qui suit le curseur comme ci-joint

```
function displayIt(e) {
  with ($("message").style) {
     left = (e.clientX - 130) + "px";
     top = (e.clientY - 25) + "px";
     visibility = "visible";
    }
}
```



Exercice 801

Exercice Faire un texte d'alerte qui suit le curseur.

```html
<!DOCTYPE html>
<html>

<head>
  <title>Test d'affichage texte pointeur</title>
  <link rel="stylesheet" href="ex801.css" type="text/css" charset="utf-8">
  <script type="text/javascript" src="ex801.js"></script>
</head>

<body>
 <h1>Test d'affichage texte pointeur</h1>
 <div id="region">
    <span id="message">Ne pas cliquer ici SVP!</span>
 </div>

</body>
</html>
```

ex801.html

```css
span#message{
    color: red;
    position: relative;
    font-size: 20pt; font-style: italic;
    font-weight: bold;
}

div#region{
    margin-left: 1in;
    margin-top: 1in;
    border: 2px solid #999;
    width: 5in;
    height: 2in;
}
```

ex801.css

Exercice 801

```
"use strict";

// ****************************************************
// The event handler function to display the message

function displayIt(evt) {
    let rgn = document.getElementById ("region");
    let r = rgn.getBoundingClientRect();
    let msg = document.getElementById("message");
    let s = msg.style;
    s.left = (evt.clientX - 130 - r.left) + "px";
    s.top = (evt.clientY - 25 - r.top ) + "px";
    s.visibility = "visible";
}
```

calcul de l'emplacement du message par rapport à la région enveloppante

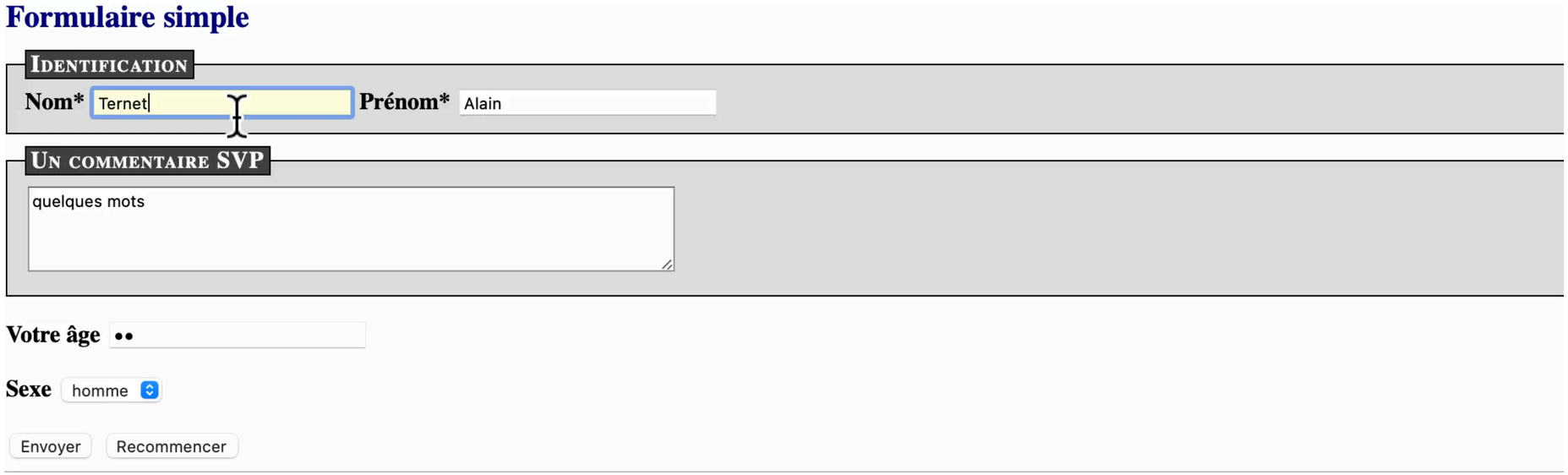
affichage du message «ne pas cliquer ici»

```
// ****************************************************
// The event handler function to hide the message

function hideIt(evt) {
   let msg = document.getElementById ("message");
   msg.style.visibility ="hidden";
}
```

```
function addEventListeners(){
    let rgn = document.getElementById ("region");
    let msg = document.getElementById ("message");
    msg.style.visibility ="hidden";
    rgn.addEventListener("mousedown", displayIt, false);
    rgn.addEventListener("mouseup", hideIt, false);
}

window.addEventListener("load", addEventListeners, false);
```

Solutions des exercices

Exercice 802 Faire la page web donnant le formulaire ci-joint



Exercice 802

- le fichier `ex802.html`

```
<!DOCTYPE html>

<html>
<head>
    <meta http-equiv="content-type" content="text/html;charset=UTF-8"/>
    <link rel="stylesheet" type="text/css" href="ex802.css"/>
    <script type="text/javascript" src="ex802.js"></script>
    <title>Un petit formulaire avec du style</title>
</head>
```

Exercice 802

- le fichier `ex802.html`

```
<!--<form action="http://www.iro.umontreal.ca/%7elapalme/echoNoURLdecode.cgi"  -->

<body>
    <h1>Formulaire simple</h1>

<form action="http://localhost/progs/ex802.php" target="_blank"
    method="get" id="formulaire" enctype="application/x-www-form-urlencoded">
    <fieldset>
        <legend>Identification</legend>
        <label class="required">Nom</label>
        <input type="text" name="nom" value="Dupont" size="24" tabindex="2">
        <label class="required">Prénom</label>
        <input type="text" name="prenom" value="Alain" size="24" tabindex="3">
        <input type="hidden" name="typeFormulaire" value="simple">
    </fieldset>
```

[la directive `tabindex` définit l'ordre de navigation avec la touche `tab`]

- le fichier `ex802.css`

```
h1 {
        font-size: 130%;
        color: navy;
        margin: 0 0 0.5em;
}

h2 {
        font-size: 115%;
        color: navy;
        margin: 0.5em 0;
}

input:focus, textarea:focus, select:focus {
        background: #ffd;
}

label{
        font-weight: bold;
}

label.required:after {
        content: '*';
}


fieldset {
        background: #ddd;
        border: 1px solid black;
        margin: 0.5em 0 0;
}

legend {
        border: 1px solid black;
        background: #444;
        color: white;
        font-variant: small-caps;
        letter-spacing: 0.1ex;
        font-weight: bold;
        padding: 0 0.2em;
}

input {
        padding: 0 0.2ex;
}

.validation {
    border: 2px solid green;
}
```

Exercice 802

- le fichier `ex802.html`

```
    <fieldset>
        <legend>Un commentaire SVP</legend>
        <textarea name="commentaire" rows="4" cols="60" tabindex="4">quelques mots SVP</textarea>
    </fieldset>
    <p>
        <label>Votre âge</label>
        <input type="password" name="age" value="-1" size="24" tabindex="5">
    </p>
    <p><label>Sexe</label>
        <select name="sexe" size="1" tabindex="6">
            <option selected="selected" value="M">homme</option>
            <option value="F">femme</option>
        </select>
    </p>
    <p><input type="submit" name="envoyer" value="Envoyer" tabindex="7">
        <input type="reset" value="Recommencer">
    </p>
</form>

</body>
</html>
```

[la directive `tabindex` définit l'ordre de navigation avec la touche `tab`]

- le fichier `ex802.css`

```
h1 {
        font-size: 130%;
        color: navy;
        margin: 0 0 0.5em;
}

h2 {
        font-size: 115%;
        color: navy;
        margin: 0.5em 0;
}

input:focus, textarea:focus, select:focus {
        background: #ffd;
}

label{
        font-weight: bold;
}

label.required:after {
        content: '*';
}


fieldset {
        background: #ddd;
        border: 1px solid black;
        margin: 0.5em 0 0;
}

legend {
        border: 1px solid black;
        background: #444;
        color: white;
        font-variant: small-caps;
        letter-spacing: 0.1ex;
        font-weight: bold;
        padding: 0 0.2em;
}

input {
        padding: 0 0.2ex;
}

.validation {
    border: 2px solid green;
}
```

Exercice 802

- le fichier `ex802.js`

```
function valide (champ, valeurInit) {
    var v = champ.value;
    if (v == valeurInit)
        window.alert("Veuillez changer le champ " + champ.name);
    else if (v.length == 0)
        window.alert("Le champ " + champ.name + " est vide");
    else return true;
    champ.focus();
    champ.select();
    return false;
}
```

```
function validate (event) {
    let f = document.getElementById("formulaire");
    if ((valide (f.nom, "Dupont") || valide (f.prenom, "Alain"))
      && valide (f.age, "-1")) {
            let a = f.age.value;
            if (a <= 0)
                window.alert("âge négatif ou nul: " + a);
            else if (a > 100)
                window.alert("âge=" + a + " Etes-vous vraiment plus que centenaire ?");
            else
                return;
            f.age.focus ();
            f.age.select ();
            return event.preventDefault ();// pour empécher la soumission
    }
    return event.preventDefault ();// pour empécher la soumission
}
```

```
function installerValidation (){
    document.getElementById ("formulaire").addEventListener ("submit", validate, false);
    document.title += " et sa validation";
}

window.addEventListener("load", installerValidation, false);
```

Exercice 802

- le fichier `ex802.php`

```php
<!DOCTYPE html>
<html>
   <head>
      <meta http-equiv="content-type" content="text/html;charset=UTF-8">
      <title>echo PHP</title>
   </head>
   <body bgcolor="#ffffff">
      <p>Voici le contenu des variables reçues via GET</p>
      <table border="1" cellspacing="2" cellpadding="3">
       <tr><td>nom = <?php echo $_GET['nom']; ?></td></tr>
       <tr><td>prénom = <?php echo $_GET['prenom']; ?></td></tr>
       </tr><td>âge = <?php echo (int)$_GET['age']; ?></td></tr>
        <tr><td>sexe = <?php echo $_GET['sexe']; ?></td></tr>
     </table>
    </body>
</html>
```

Fichiers dans cgi-bin

```
<html>

<body>

<form method="get" action="http:/localhost/progs/cgi-bin/test.py" target="_blank">

Red<input type="checkbox" name="color" value="red">

Green<input type="checkbox" name="color" value="green">

<input type="submit" value="Submit">

</form>
</body></html>
```

fichier `test.py` dans un répertoire cgi-bin sur le serveur ici `localhost`

```
#!/usr/bin/env python
import cgi
form = cgi.FieldStorage()

# getlist() returns a list containing the
# values of the fields with the given name
colors = form.getlist('color')

print ("Content-Type: text/html\n")
print ('<html><body>')
print ('The colors list:', colors)
for color in colors:
    print ('<p>', cgi.escape(color), '</p>')
print ('</body></html>')
```

HTML ++

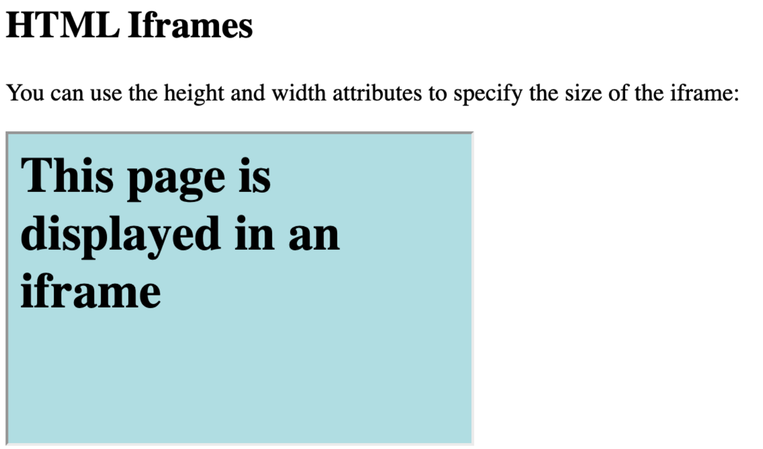
- on peut insérer une page web dans un cadre <iframe>

```
<iframe src="url" title="description"></iframe>
```

- et on peut activer une ancre pour insérer dans un cadre

```
<iframe src="demo_iframe.htm" name="iframe_a" title="Iframe Example"></iframe>

<p><a href="https://www.w3schools.com" target="iframe_a">W3Schools.com</a></p>
```

- voir http://www.w3schools.com/html/html_iframe.asp
  pour une description détaillée



HTML++

- les URL (*Uniform Resource Locator*) adressent des documents

  `http` (*HyperText Transfer Protocol*)

  `https` (*Secure HyperText Transfer Protocol*)

  `file` pour les fichiers locaux

- les URL ne contiennent que des caractères ASCII

  `espace` s'écrit `%20` `(code ASCII 32 en hexadécimal)`

HTML++

- les blocs (*blocks*) font un saut de ligne et ont une marge au-dessus et en-dessous

`<address>`	`<article>`	`<aside>`	`<blockquote>`	`<canvas>`	`<dd>`
`<div>`	`<dl>`	`<dt>`	`<fieldset>`	`<figcaption>`	`<figure>`
`<footer>`	`<form>`	`<h1>`–`<h6>`	`<header>`	`<hr>`	`<li>`
`<main>`	`<nav>`	`<noscript>`	`<ol>`	`<p>`	`<pre>`
`<section>`	`<table>`	`<tfoot>`	`<ul>`	`<video>`	

- les éléments en ligne (*inline*) sont simplement insérés dans le texte

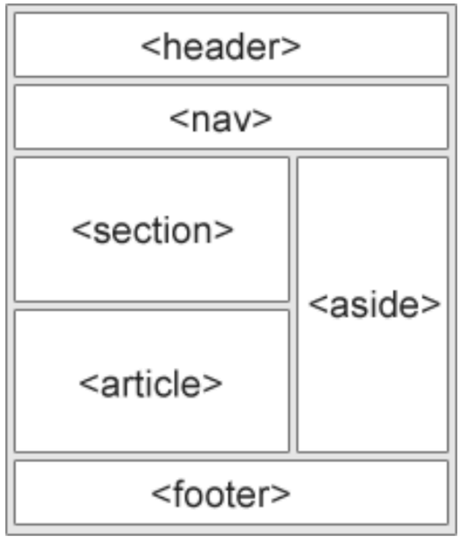
`<a>`	`<abbr>`	`<acronym>`	`<b>`	`<bdo>`	`<big>`
` `	`<button>`	`<cite>`	`<code>`	`<dfn>`	`<em>`
`<i>`	`<img>`	`<input>`	`<kbd>`	`<label>`	`<map>`
`<object>`	`<output>`	`<q>`	`<samp>`	`<script>`	`<select>`
`<small>`	`<span>`	`<strong>`	`<sub>`	`<sup>`	`<textarea>`
`<time>`	`<tt>`	`<var>`			

HTML++

- les dispositions (*layout*) organisent la géométrie d'une page

CSS *frameworks* CSS flexbox CSS *float property* CSS *grid*

- exemple:



- `<header>` - Defines a header for a document or a section
- `<nav>` - Defines a set of navigation links
- `<section>` - Defines a section in a document
- `<article>` - Defines an independent, self-contained content
- `<aside>` - Defines content aside from the content (like a sidebar)
- `<footer>` - Defines a footer for a document or a section
- `<details>` - Defines additional details that the user can open and close on demand
- `<summary>` - Defines a heading for the `<details>` element

- voir `http://www.w3schools.com/html/html_layout.asp`

  pour une description détaillée

W3-CSS

- un paquetage de styles W3 (*World Wide Web*)

voir `http://jeanjacqueslevy.net/courses/21eaaa/`

voir `http://jeanjacqueslevy.net/courses/21beaaa/`

- voir `http://www.w3schools.com/w3css/default.asp`

  pour une description détaillée

JavaScript ++

- champs *getter* et *setter* dans un objet

```
const student = {
    firstName: 'Monica',

    // accessor property(getter)
    get getName() {
        return this.firstName;
    }
};

// accessing data property
console.log(student.firstName); // Monica

// accessing getter methods
console.log(student.getName); // Monica

// trying to access as a method
console.log(student.getName()); // error
```

```
const student = {
    firstName: 'Monica',

    //accessor property(setter)
    set changeName(newName) {
        this.firstName = newName;
    }
};

console.log(student.firstName); // Monica

// change(set) object property using a setter
student.changeName = 'Sarah';

console.log(student.firstName); // Sarah
```

JavaScript ++

- définir une propriété par programme: `defineProperty`

```
const student = {
    firstName: 'Monica'
}

// getting property
Object.defineProperty(student, "getName", {
    get : function () {
        return this.firstName;
    }
});

// setting property
Object.defineProperty(student, "changeName", {
    set : function (value) {
        this.firstName = value;
    }
});

console.log(student.firstName); // Monica

// changing the property value
student.changeName = 'Sarah';

console.log(student.firstName); // Sarah
```

JavaScript ++

- étendre un objet (indépendamment des classes): `prototype`

  [*deprecated*: il vaut mieux utiliser les classes]

```
/ constructor function
function Person () {
    this.name = 'John',
    this.age = 23
}

// creating objects
const person1 = new Person();
const person2 = new Person();

// adding property to constructor function
Person.prototype.gender = 'male';

// prototype value of Person
console.log(Person.prototype);

// inheriting the property from prototype
console.log(person1.gender);
console.log(person2.gender);
```

Prochain cours

- un bon tutoriel JavaScript: `http://www.programiz.com/javascript`
- un autre tutoriel JavaScript: `http://www.w3schools.com/js`
- modules, prototypes, `getter`, `setter` en JavaScript
- quelques bibliothèques JavaScript
- protocole HTTP